https://doi.org/10.1007/s11704-025-41126-5

RESEARCH ARTICLE

Patching the visual ability of large multimodal models by collaborating with small models

Hao LIANG[1,2], Xiaolong ZHANG[1,2], Meina KAN[1,2]✉, Shiguang SHAN[1,2,3], Xilin CHEN[1,2]

1. State Key Laboratory of AI Safety, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
2. University of Chinese Academy of Sciences, Beijing 100049, China
3. Peng Cheng National Laboratory, Shenzhen 518055, China

Received October 20, 2024; accepted May 20, 2025
E-mail: kanmeina@ict.ac.cn
© Higher Education Press 2026

Abstract
Large multimodal models (LMMs) have demonstrated significant success across various tasks but fall short on some basic visual functions, such as inaccurate object counting and imprecise localization. These limitations restrict the application of LMMs in broad scenarios. To enhance the capabilities of LMMs, we propose a novel method to patch their visual perceptual abilities by collaborating with small task-specific models. Our method begins with utilizing an LMM to decompose the user query into a series of visual functions. For each function, the appropriate model, either the LMM itself or a small task-specific model, is invoked. To determine whether to patch the LMM with a small task-specific model, we design a novel question-answering-based reinforcement learning strategy to optimize the decision process. Finally, the LMM generates the answer utilizing the visual perceptual results. The proposed method is evaluated on two standard visual question-answering datasets and two specialized datasets. The experimental results demonstrate that our method effectively enhances the visual abilities of LMMs.

Keywords
model collaboration; patching visual ability; large multimodal models

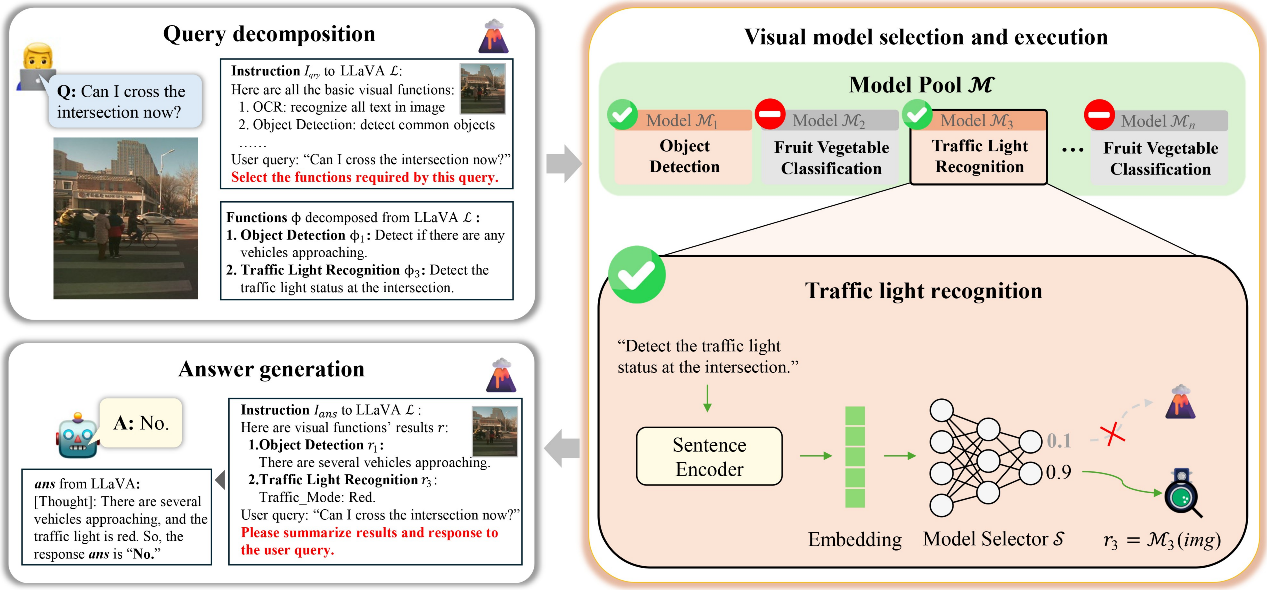
1 Introduction

Recently, advancements in large multimodal models (LMMs) [1,2] have yielded remarkable achievements in diverse fields such as visual question answering, programming, and gameplay, marking progress toward artificial general intelligence (AGI). LMMs excel in perception, reasoning, and planning, making them adept at tracking complex tasks, such as controlling robots and programming. Among these abilities, visual perception plays a crucial role, akin to the role of the eye in humans. It acts as a foundational skill, providing essential information for reasoning and planning.

Nevertheless, when focusing only on the visual ability of LMMs, it is clear that they exhibit certain deficiencies, such as imprecise object localization, difficulty in accurate counting, and inaccurate text recognition [3,4]. Despite the identification of a few shortcomings in the visual ability of LMMs, addressing these issues is quite challenging due to the complex training processes and the massive data used by LMMs. Furthermore, completely resolving all limitations seems nearly impossible, as these shortcomings tend to be identified sequentially during usage, suggesting a potentially endless cycle of discovery and amendment.

To overcome these deficiencies, strategies from large language models (LLMs) can offer valuable insights. For instance, LLMs have successfully extended their functionalities by integrating external tools or models. For example, Visual ChatGPT [5] and Toolformer [6] enhance LLMs by integrating external tools and models. However, these methods differ in the case of LMMs, as they primarily augment capabilities that LLMs lack, whereas LMMs already possess strong visual abilities but may still perform poorly in certain cases.

To specifically enhance the visual capabilities of LMMs, we propose a novel approach that patches their visual ability by collaborating with small task-specific models. This strategy allows for the continuous improvement of the visual capabilities of LMMs, addressing deficiencies in a targeted and efficient manner. Specifically, our method begins by utilizing an LMM to decompose the user query into a series of visual functions. For each function, the appropriate model, either the LMM itself or a small task-specific model, as a patch, is invoked via a model selector. Ultimately, the LMM generates the answer utilizing these visual perceptual results. Figure 1 illustrates the workflow of our method. In this procedure, the primary challenge lies in how to obtain an accurate model selector, which involves assessing the performance of both the LMM and the smaller models and thus determining to choose a superior one for a visual function. To address this, we propose a novel question-


Query decomposition
Q: Can I cross the intersection now?
Visual model selection and execution
Model Pool $\mathcal{M}$
Model $\mathcal{M}_1$
Object Detection
Model $\mathcal{M}_2$
Fruit Vegetable Classification
Model $\mathcal{M}_3$
Traffic Light Recognition
…
Model $\mathcal{M}_n$
Fruit Vegetable Classification
Traffic light recognition
Sentence Encoder
Embedding
Model Selector $\mathcal{S}$
0.1
0.9
$r_3 = \mathcal{M}_3(img)$
Answer generation
A: No.

Instruction I_{qry} to LLaVA $\mathcal{L}$:
Here are all the basic visual functions:
1. OCR: recognize all text in image
2. Object Detection: detect common objects
……
User query: “Can I cross the intersection now?”
Select the functions required by this query.

Functions ϕ decomposed from LLaVA $\mathcal{L}$ **:**
1. Object Detection ϕ_1 **:** Detect if there are any vehicles approaching.
2. Traffic Light Recognition ϕ_3**:** Detect the traffic light status at the intersection.

“Detect the traffic light status at the intersection.”

Instruction I_{ans} to LLaVA $\mathcal{L}$:
Here are visual functions’ results r:
1.Object Detection r_1 **:**
There are several vehicles approaching.
2.Traffic Light Recognition r_3 :
Traffic_Mode: Red.
User query: “Can I cross the intersection now?”
Please summarize results and response to the user query.

ans from LLaVA**:**
[Thought]: There are several vehicles approaching, and the traffic light is red. So, the response ***ans*** is “**No.**”

Fig. 1 An illustration of our method. First, the user’s query is decomposed into visual functions. Then, our proposed model selector determines whether to use the LMM or a small task-specific model for each function based on the generated descriptions. Finally, the LMM synthesizes the results from all visual functions to generate the final answer

answering-based reinforcement learning strategy for optimizing the decision-making process.

To summarize, our main contributions are:

- We propose a novel method to patch the visual ability of LMMs by collaborating with small task-specific models.
- We introduce a question-answering-based reinforcement learning strategy to train an effective model selector that can determine whether to choose the large or small models based on the task at hand.
- The evaluations on both standard and specialized datasets demonstrate that our method can significantly improve the visual abilities of large multimodal models.

2 Related work

LMMs have been widely studied in recent years, with many general-purpose models designed to handle a broad range of tasks. These models, however, often face limitations in specialized visual functions. To address these challenges, various strategies have been proposed, including augmenting LMMs with external tools and developing models tailored for specific applications.

2.1 General-purpose large multimodal models

The development of general-purpose models capable of handling diverse tasks has been a key focus in artificial intelligence. The success of ChatGPT [7], an LLM based on the generative pre-trained transformer architecture, demonstrated not only its effectiveness in tasks like text generation, summarization, and question answering but also underscored the potential path toward AGI. This inspired subsequent works such as LLaMA [8,9], ChatGLM [10], and Vicuna [11], which further advanced LLM capabilities. These advancements have paved the way for extending LLMs to multimodal settings, enabling models to process information across diverse modalities like vision and audio.

In order to equip LLMs with visual capabilities, researchers have proposed methods to align image features with textual representations. Works like Flamingo [12], GPT-4V [2], and MiniGPT [13] extract image features using encoders such as Vision Transformers [14] and project them into the textual embedding space [15], enabling tasks like image captioning [1] and visual reasoning [16]. Similarly, VideoChat [17] and LLaMA-VID [18] extend these capabilities to video inputs, supporting tasks like video summarization and conversational analysis. Beyond vision, frameworks like AudioPaLM [19] and IMAGEBIND [20] integrate audio and other modalities, achieving strong zero-shot performance across diverse domains.

Despite these advancements, general-purpose models often struggle with tasks requiring fine-grained visual perception or domain-specific expertise, such as precise object detection or text recognition. This limitation is particularly critical for applications like assistive technologies for visually impaired individuals. Our method addresses this gap by dynamically integrating LMMs with task-specific models, leveraging their complementary strengths to achieve robust performance across a wider range of tasks.

2.2 Large models use multimodal tools

Although general-purpose LMMs have achieved remarkable performance, they tend to underperform in some basic abilities [3,4] and struggle with improving through limited training data. Consequently, another research direction has emerged within the AI community: teaching LLMs to utilize tools [6,21–23], such as leveraging external search engines to access the latest data [22], enhancing computational abilities with calculators [6], incorporating image editing and generation models [24], and utilizing Python interpreters for improved code generation [23]. These external APIs and tools significantly enhance the capabilities of LLMs.

These methods typically involve three stages: task decomposition, model execution, and answer generation. Representative works include MM-REACT [25], Visual ChatGPT [5], HuggingGPT [21], ControLLM [26], and Toolformer [6]. Task decomposition is often performed by LLMs such as ChatGPT [7] and GPT-4 [2], which break down complex tasks into smaller, more manageable subtasks to identify the necessary tools. For example, HuggingGPT [21] employs descriptions sourced from the *Hugging Face* Model Hub as inputs for ChatGPT [7], while ControlLLM [26] utilizes a Thoughts-on-Graph strategy, which explores the optimized solution through a tool graph capturing tool dependencies and parameters. Then, the selected tools are employed in the model execution stage. In the final stage, answer generation, the intermediate results are summarized by LLMs to generate the final response. Most of these methods require fine-tuning the entire LLM, which implies the necessity of gathering diverse data and conducting multiple experiments to ensure that the LLM evolves in the desired direction. In contrast, our method does not require fine-tuning the large model, making it more expedient and efficient. Additionally, our method preserves the inherent capabilities of the large model by avoiding fine-tuning.

Besides these methods, LLaVA-Plus [27] most closely aligns with our work, with both targeting the enhancement of visual abilities in LMMs. However, they differ in how to improve their visual abilities. LLaVA-Plus [27] is designed to augment the visual capabilities that an LMM lacks, whereas our method focuses on strengthening the visual capabilities that the LMMs already possess. Furthermore, our approach does not simply lean towards smaller models but chooses the model given the current user query, thus creating a synergy between large and small models.

Beyond image-based tasks, DoraemonGPT [28] leverages LLMs and task-specific tools for dynamic video understanding. In contrast, our method focuses on image-based tasks and modularizes the task-solving process by dynamically selecting between an LMM and smaller specialized models. Recent studies, such as the Multiple Knowledge Representation (MKR) framework [29], also highlight the value of integrating diverse knowledge sources to improve the generalization and explainability of AI systems. Neural Architecture Search (NAS) [30,31] also involves the exploration of different sub-networks to optimize the overall performance, but it does so in a static manner. In contrast, our method dynamically integrates an LMM with smaller task-specific models during inference.

2.3 Large models for special usage

In efforts to enhance the model’s capabilities on specific tasks, significant improvements have been shown through fine-tuning with instruction-following datasets, as indicated by the studies of [32] and [33]. Moreover, Self-Instruct [34] demonstrates that LLMs can improve their performance through instruction fine-tuning on their own generated data, thus reducing the need for human-written instructions. Additionally, many works have been dedicated to adapting large models for specific fields [35–37]. For instance, Med-PaLM [35] is trained on both domain-specific and general data, which significantly improves its capabilities in the medical sector while maintaining its general proficiency. Similarly, Galactica [36], after being trained on millions of scientific data, has developed the ability to store, combine, and reason about scientific knowledge. All these methods involve fine-tuning the Large Models (LMs) for a specific task, which may compromise other capabilities of the LMs. In contrast, our method does not fine-tune the LM and also retains the original proficiency.

3 Method

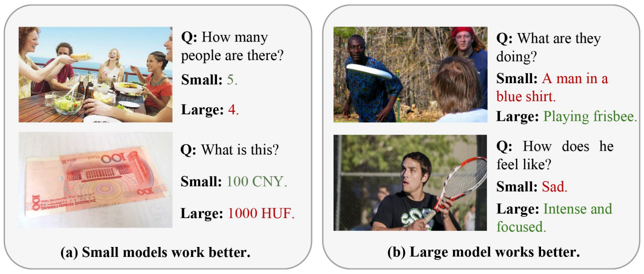
The LMMs have achieved great success in various tasks, benefiting from their comprehensive abilities in perception, understanding, and reasoning. Nevertheless, they sometimes fall short in basic visual functions such as object localization and counting, as shown in Fig. 2. To address this issue, we proposed a method to patch the visual ability of LMMs through collaboration with small task-specific models. Our research is conducted in the form of Visual Question Answering (VQA), which is a common task that fully leverages the capabilities of LMMs.

Formally, given a user query qry and a corresponding image img, our method enhances the visual ability and answer quality of the LMM $\mathcal{L}$. Instead of using $\mathcal{L}$ alone, we augment it with a set of smaller models $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$.

As illustrated in Fig. 1, our method consists of three parts, including query decomposition, model selection and execution, and answer generation. *Firstly*, given the query qry and the image img, the LMM $\mathcal{L}$ is used to decompose the capabilities necessary for handling this query, leading to one set of visual functions $\Phi = \{\phi_1, \ldots, \phi_n\}$. *Secondly*, for each visual function ϕ_i, a model selector $\mathcal{S}$ is trained to choose the LMM $\mathcal{L}$ itself or a small task-specific model $\mathcal{M}_i$ to execute. This process leads to a group of chosen visual models $\{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$, and the execution of these selected models yields a set of visual perceptual results $R = \{r_1, \ldots, r_n\}$. *Finally*, the LMM $\mathcal{L}$ generates an answer ans by using the user query qry, input image img, and visual perceptual results R as input. The subsequent subsections provide a detailed description of each component.

3.1 Query decomposition

In the context of the VQA task, a user query usually involves multiple aspects such as perception, understanding, and reasoning. In



(a) Small models work better. **(b) Large model works better.**

Fig. 2 An exemplar illustration of strengths and weaknesses in various capabilities between small and large models. The small models perform better in the left examples, while the large model performs better in the right examples

this work, our focus is on enhancing the visual perception capabilities of LMMs. Therefore, the first step of our method is to decompose the query into a series of visual functions $\Phi = \{\phi_1, \ldots, \phi_n\}$, which will later be executed in the model selection and execution phase.

$$\Phi = \mathcal{L}(qry,\ img;\ I_{qry}). \tag{1}$$

As shown in Eq. (1), $\mathcal{L}$ represents an LMM (e.g., LLaVA [1], GPT4-V [2]). To facilitate the query decomposition for subsequent processing, we design an instruction I_{qry} that guides $\mathcal{L}$ in generating a set of visual functions Φ in JSON format. Each function ϕ_i consists of a name and a target description. *A specific example of a visual function ϕ_i includes having "Object Detection" as the function name, with the target description being "To locate vehicles using object detection"*. The detailed I_{qry} used for decomposition is based on MM-REACT [25] and provided in the Appendix.

3.2 Visual model selection and execution

In this stage, the visual functions should be implemented as specific models. The primary goal is to identify the better one of the LMM and the small task-specific model for each visual function ϕ_i.

In previous works such as Toolformer [6] and GPT4Tools [24], this phase is not included, as there is only one small task-specific model that serves as the candidate for each function, so results from task decomposition can directly coordinate it. However, in our method, we introduce a suite of small models $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$ to patch the visual capabilities that the LMM may underperform. Consequently, this introduces two potential candidates for executing each visual function: a model from the suite or the LMM itself, represented as $\{M_i, \mathcal{L}\}$. Therefore, a decision-making process is required to select the more appropriate model for each function. To make this selection, we propose a model selector $\mathcal{S}$ to choose one of the large and small models depending on their relative performance on this function, formulated as follows:

$$N_s = \mathcal{S}(\phi_i), \tag{2}$$

where the input ϕ_i contains the function name and the target description. The model selector $\mathcal{S}$ is implemented as a simple neural network, including a pre-trained sentence encoder MiniLM [38] and several fully connected layers. The output N_s is a real score within the range $[0, 1]$. So, the final model $\mathcal{F}_i$ selected to implement the visual function ϕ_i is:

$$\mathcal{F}_i = \begin{cases} \mathcal{M}_i, & \text{if } N_s \geqslant 0.5, \\ \mathcal{L}, & \text{otherwise.} \end{cases} \tag{3}$$

Once a model has been selected, it is executed to produce the intermediate result r_i as follows:

$$r_i = \begin{cases} \mathcal{M}_i(img), & \text{if } \mathcal{F}_i = \mathcal{M}_i, \\ \mathcal{L}(img,\ I_{\phi_i}), & \text{otherwise.} \end{cases} \tag{4}$$

During the execution phase, if the small visual model $\mathcal{M}_i$ is chosen, it is directly operated using the image img as input. Conversely, if the LMM $\mathcal{L}$ is selected, a tailored prompt I_{ϕ_i} is employed to facilitate the execution of the function ϕ_i via the LMM $\mathcal{L}$. The design of the prompt I_{ϕ_i} is detailed in the Appendix. Finally, when models for all functions are chosen and executed, we obtain the final visual perceptual results as below:

$$R = \{r_1, r_2, \ldots, r_n\}. \tag{5}$$

The model selector is optimized using a reinforcement learning strategy, which is detailed in Sec. 3.4.

3.3 Answer generation

With the visual perceptual results from Eq. (4), the LMM $\mathcal{L}$ finally generates an answer $\widehat{ans}$ for the user query by taking the visual perceptual results R, the image img, and the user's query qry as input:

$$\widehat{ans} = \mathcal{L}(img,\ I_{ans}(qry, R)), \tag{6}$$

where an instruction I_{ans} is designed to generate a user-friendly answer by converting the user query qry and perceptual results R into a text prompt. Further details can be found in the Appendix.

3.4 Model selector optimization with reinforcement learning

Within the entire pipeline of our method, the only component requiring optimization is the model selector $\mathcal{S}$. For each function, the model selector $\mathcal{S}$ is expected to be able to evaluate the performance of the small model $\mathcal{M}_i$ and the LMM $\mathcal{L}$ on this function ϕ_i and make the optimal choice.

A direct approach to optimizing the model selector is assessing the performance of these models on one or several datasets with diverse visual labels. However, given the wide range of visual functions, this strategy might be time-consuming and labor-intensive. Consequently, we instead leverage a question-answering dataset and design a question-answering-based reinforcement learning strategy for optimizing the model selector.

In reinforcement learning, an agent interacts with the environment, continuously taking actions, receiving rewards, and optimizing itself. The goal is to learn how to take action (i.e., action policy) to maximize the reward it can receive, which aligns closely with our model selector. Typically, the action policy is modeled as a probability distribution of taking actions given a certain state.

Similarly, we view our question-answering process as a process of interaction between the model selector (akin to the agent) and the answer generation part (akin to the environment). As shown in Fig. 3, at the beginning of each simulation in the reinforcement learning, a random question-answer sample (qry, ans) is selected. Then, the visual functions $\Phi = \{\phi_1, \ldots, \phi_n\}$ are obtained through query decomposition. Following this, the model selector will take action a_i according to the policy $\pi_{\mathcal{S}}$ for a given visual function ϕ_i. Since there are only two choices, the policy forms a Bernoulli distribution as below:

$$\pi_{\mathcal{S}}(a_i, \phi_i) = \begin{cases} \mathcal{S}(\phi_i), & \text{if } a_i \text{ means "use } \mathcal{M}_i\text{"}, \\ 1 - \mathcal{S}(\phi_i), & \text{if } a_i \text{ means "use } \mathcal{L}\text{"}. \end{cases} \tag{7}$$

The actions continue iteratively until all visual functions have been assigned a model and executed. Then, the LMM $\mathcal{L}$ generates an answer $\widehat{ans}$. Finally, the generated answer is compared to the ground truth answer via ChatGPT [7], to generate a score akin to a reward, as below:

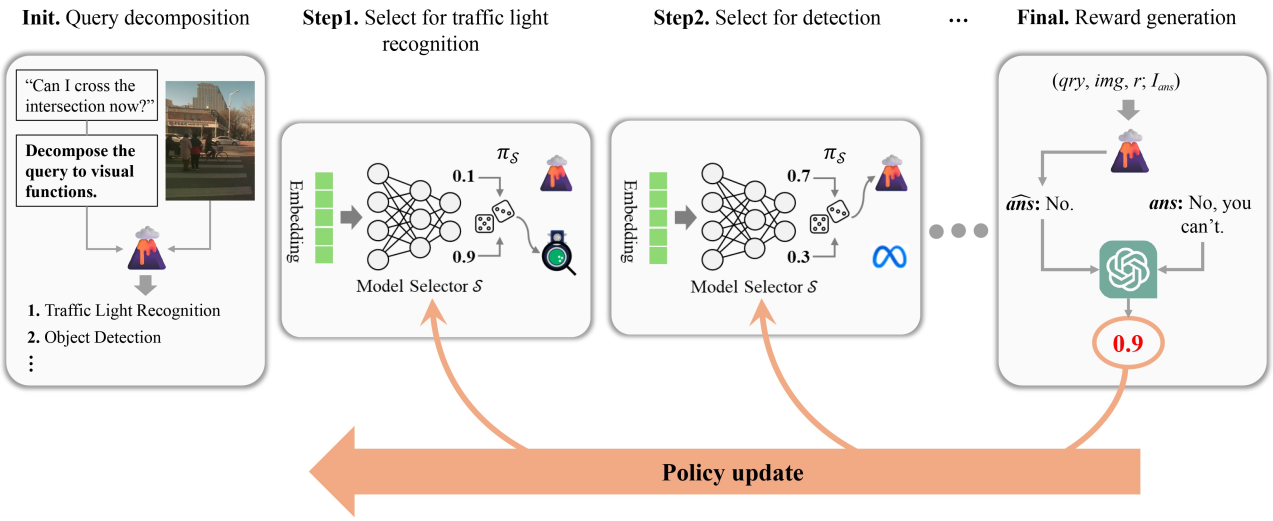

Init. Query decomposition
Step1. Select for traffic light recognition
Step2. Select for detection
…
Final. Reward generation
"Can I cross the intersection now?"
Decompose the query to visual functions.
1. Traffic Light Recognition
2. Object Detection
⋮
Embedding
0.1
0.9
π_S
Model Selector S
0.7
0.3
(qry, img, r; I_{ans})
$\widehat{ans}$: No.
ans: No, you can't.
0.9
Policy update

Fig. 3 An illustration of the training process of our question-answering-based reinforcement learning. The model selector operates as an agent, choosing a model for each function. The final response is then evaluated by ChatGPT, which provides a score. The score serves as the reward for refining the policy of the model selector

$$Reward(\widehat{ans}, ans, qry) = \mathrm{GPT}(\widehat{ans}, ans, qry, I_{score}), \tag{8}$$

where qry and ans are the sampled question and answer, $\widehat{ans}$ is the generated answer though policy π_S, I_{score} is the instruction used to guide ChatGPT [7] in generating scores, which is listed in Appendix.

By considering the model selector as an agent, its optimization process is transformed into identifying the optimal policy in reinforcement learning, which can be obtained by maximizing the expected value of the received reward:

$$\arg\max_{\pi_S} \mathbb{E}\left[Reward(\widehat{ans}, ans, qry)\right]. \tag{9}$$

It is a standard reinforcement learning task where the PPO algorithm [39] is employed to find the optimal policy π_S^*, i.e., the optimal parameters of S.

4 Experiments

To evaluate the effectiveness of our method, we conducted experiments across four VQA datasets. LLaVA-Bench [40] and MM-VET [41] serve as standard benchmarks, whereas VizWiz-VQA [42] and our own Blmm are designed to assess visual perception capabilities in scenarios involving blind users, which emphasize real-world challenges in visual perception.

In Sec. 4.1, we present our evaluation benchmarks and detail the training process. Subsequently, in Sec. 4.2, we evaluate our method's performance across various datasets by comparing it to other state-of-the-art (SOTA) methods. Finally, in Sec. 4.3, we conduct an ablation analysis to verify the effectiveness of our newly proposed modules.

4.1 Experimental setup

4.1.1 Datasets

The LMMs possess extensive capabilities in perception, understanding, reasoning, etc. Therefore, we explore the impact of our method on the overall performance of LMMs using standard VQA datasets, including LLaVA-Bench [40] and MM-VET [41].

LLaVA-Bench [40] comprises 30 images randomly chosen from the COCO validation dataset, accompanied by 90 generated questions. The questions encompass conversation, complex reasoning, and detailed descriptions.

MM-VET [41] is designed to assess the multimodal capabilities of LMMs, such as solving mathematical problems on a blackboard, which involves OCR, location identification, and calculations. The dataset includes 200 images and 218 questions, which evaluate six key vision-language abilities and their integration.

The two standard VQA datasets cover a wide range of abilities, including mathematics, logic, and reasoning, but are not tailored to evaluate specific visual functions. To address this, we introduce two perception-oriented datasets that better reflect performance across diverse visual functions.

Vizwiz-VQA [42] dataset is designed to assist the visually impaired, containing questions based on real requests and focusing on visual perception challenges. As noted in its description [42], many samples are unanswerable due to 'low-quality images and fingers blocking the camera.' Even humans cannot extract enough information from these images to answer the questions. This leads to the dominance of language priors in LMMs. To mitigate bias from these priors, we removed unanswerable samples, leaving 537 usable ones for evaluation.

Blmm dataset, collected by us, is specifically designed for visually impaired individuals, similar to VizWiz, but focuses more on assessing visual abilities. The test split contains 195 samples of visual questions and answers, mainly drawn from real-life scenarios faced by visually impaired individuals.

4.1.2 Training details

For the small models, we identified seven types of daily activities in the lives of visually impaired individuals, resulting in the development of 13 specialized visual functions. These visual functions are listed in Table 1, while the corresponding small task-specific models are provided in the Appendix. Although these visual functions may not cover all aspects of the lives of the visually

Table 1 Overview of daily activities and essential visual functions for the visually impaired

Daily activity	Basic visual functions
General	Object detection, semantic segmentation, visual grounding, image caption
Working	Cash recognition
Reading	Optical character recognition
Socialisation	Face expression recognition
Recreation	Action recognition
Diet	Dish recognition, fruit vegetable recognition
Outing	Tactilepaving segmentation, traffic light recognition, outdoors semantic segmentation

impaired, they are sufficiently representative to validate our proposed method. *For the LMM*, LLaVA-7B [1] is chosen considering its open-source nature and versatile multimodal capabilities. *For the model selector*, its architecture is established as a sentence encoder with a 384-dimensional output, followed by three fully connected layers with a hidden dimension of 64, and utilizes a Tanh activation function. *For the optimization of reinforcement learning*, the commonly used PPO algorithm [39] is employed to optimize our model selector. *For the training process*, the total number of training samples is 20,000, with a learning rate of 0.00025 and a batch size of 32. *For evaluation*, the ChatGPT [7] is exploited to compare the generated answer and the ground truth answer, adhering to the standard setting on each dataset. *For the training data*, we utilized a mixed dataset, including 452 samples from the Blmm train split, 3296 samples from the Vizwiz-VQA train split [42], and 100 samples from the VQA2.0 dataset [43].

4.2 Comparisons with existing methods

This subsection evaluates the efficacy of our method through a comparative analysis with recent methods across four datasets. LLaVA [1] serves as the baseline, representing an independent LMM. GPT4Tools [24] and VisGPT [5] integrate tools into LLMs, while ControlLLM [26] integrates multimodal tools into the large language model LLaMA-7B, with LLaVA-7B also integrated as a tool. LLaVA-Plus [27] integrates tools into LLaVA-7B, which is the most closely related to our method. However, unlike LLaVA-Plus, which fine-tunes LLaVA to teach it how to use multimodal tools, our method requires no fine-tuning of LLaVA. Instead, we only train a lightweight model selector that learns to efficiently choose between large and small models. Several samples and visualized results are presented in the Appendix C.

4.2.1 Results on LLaVA-Bench and MM-VET

Firstly, all methods are evaluated using the LLaVA-Bench and MM-VET to investigate the comprehensive capabilities of LMMs. The comparative results are shown in Table 2. For LLaVA-Plus, we used the results reported in their official paper; for other methods, since they had not been evaluated on these datasets, we used their official code for evaluation. As shown in Table 2, our method significantly outperforms the baseline model LLaVA [1], with improvements of 3.9% on LLaVA-Bench and 7.0% on MM-VET, respectively. Since our method shares the same LMM as LLaVA, the performance gain primarily comes from the introduced small models, highlighting their effectiveness. Furthermore, our method surpasses two vanilla baselines: Ours($\mathcal{L}$) (which always chooses the large model) and Ours($\mathcal{M}$) (which always chooses the small model). This superiority underscores the effectiveness of the model selection strategy learned through the reinforcement learning algorithm. It is also worth noting

Table 2 Comparative performance of various LMMs on LLaVA-Bench and MM-VET. All methods are based on LLaVA-7B, LLaMA-7B, or ChatGPT. * indicates that the model does not utilize LMMs

	LLaVA-Bench				**MM-VET**						
	Conv.	Detail	Reas.	All	Rec	OCR	Know	Gen	Spat	Math	All
LLaVA	73.6	66.1	76.3	72.5	30.4	13.3	19.2	20.1	18.7	8.1	24.1
LLaVA-Plus	–	–	–	–	30.5	23.6	**20.5**	**22.5**	**28.5**	7.7	27.5
CLLM	71.4	66.5	**83.9**	74.4	20.3	10.2	6.2	7.4	19.2	0.0	16.7
VisGPT*	61.5	40.9	59.3	54.5	21.2	9.0	6.9	6.1	12.8	3.5	17.1
GPT4Tools*	72.6	42.9	82.4	67.0	20.9	8.6	7.2	7.3	15.1	0.0	16.4
Ours($\mathcal{L}$)	76.3	**72.4**	67.0	71.9	28.2	20.2	15.6	14.9	20.5	7.3	25.8
Ours($\mathcal{M}$)	72.2	67.1	83.3	74.7	30.2	20.5	11.8	14.4	22.9	7.7	26.7
Ours	**77.2**	71.5	83.0	**76.4**	**32.1**	**29.3**	20.1	22.4	27.3	**15.4**	**31.1**

that GPT4Tools [24] and VisGPT [5], which integrate visual tools into LLMs, show marginally lower performance. This can be attributed to LLMs' inherent lack of visual capabilities, i.e., LLMs depend on small models to acquire visual information.

4.2.2 Results on VizWiz and Blmm

Secondly, all methods are evaluated on the VizWiz and Blmm datasets, which specifically focus on assessing visual ability. The comparative results are shown in Table 3. From these results, we observe a similar conclusion that our method performs better than those LMMs as well as the vanilla baselines. On these two datasets, our method achieves larger improvement, 5.6% and 7.8%, compared to LLaVA on VizWiz and Blmm, respectively. This further demonstrates that our method can effectively enhance the visual abilities of LMMs.

Table 3 Comparative performance of various LMMs on VizWiz and Blmm. All methods are based on LLaVA-7B, LLaMA-7B, or ChatGPT. "OCR" and "Det." stand for Optical Character Recognition and Object Detection, respectively. These are the most frequently used visual functions in the VizWiz and Blmm datasets

	VizWiz			Blmm		
	OCR	Det.	All	OCR	Det.	All
LLaVA	71.5	67.0	70.8	54.0	50.8	54.8
LLaVA-Plus	21.5	14.9	21.2	32.1	36.9	37.5
CLLM	43.2	41.8	43.1	42.4	53.1	46.8
VisGPT*	40.8	39.6	39.8	40.8	39.6	39.9
GPT4Tools*	42.9	34.9	42.9	41.0	50.2	47.0
Ours($\mathcal{L}$)	72.6	69.4	71.5	51.0	52.7	54.6
Ours($\mathcal{M}$)	76.7	73.2	75.7	**59.6**	59.4	61.7
Ours	**78.7**	**73.9**	**76.4**	59.5	**61.7**	**62.6**

4.3 Ablation study

4.3.1 Impact of model selection strategy

To investigate the necessity and effectiveness of the model selection, we compare the performance of several different strategies for model selection: always selecting the large model, always selecting the small model, randomly selecting, and selecting using our model selector. The results in Fig. 4 demonstrate that our method outperforms other strategies in nearly every visual function, which clearly demonstrates the exceptional performance of our method on individual visual functions.

4.3.2 Correlation between the selection strategy and the model's performance

To identify whether the strategy employed by our model selector accurately represents the utility of large and small models, we show the probability of using small models generated by our model

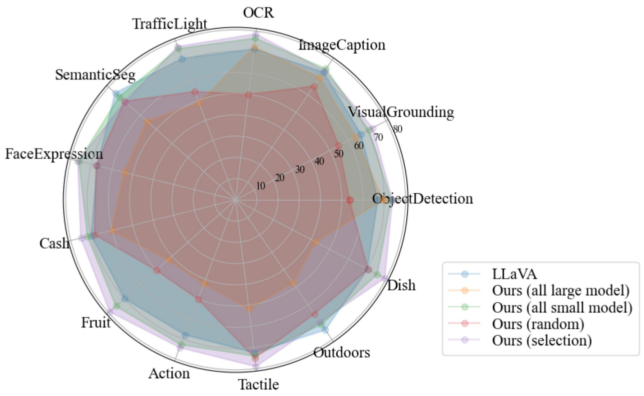
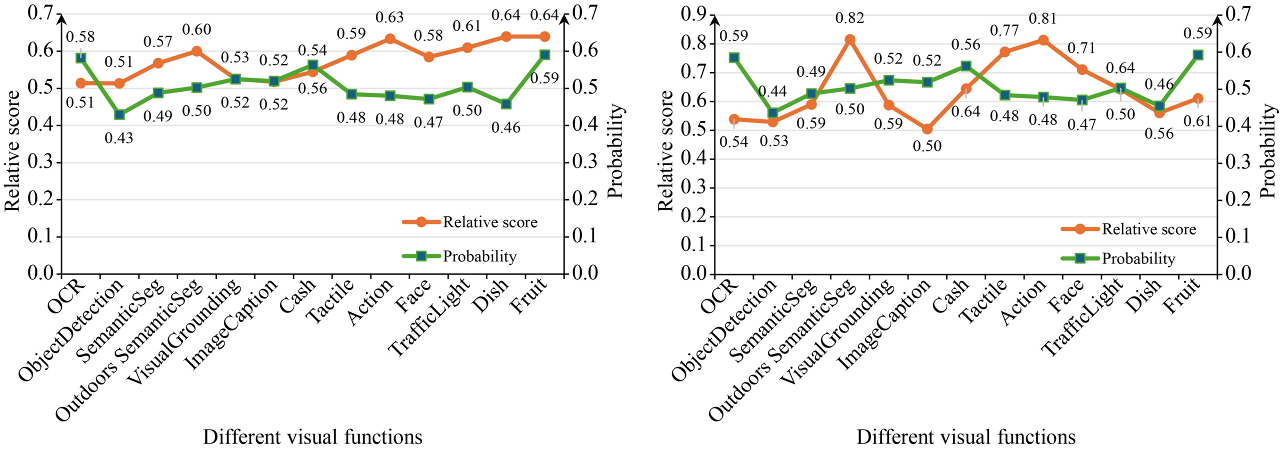


Fig. 4 Ablation study on the impact of various model selection strategies across all utilized visual functions on the VizWiz dataset. The LLaVA establishes a baseline. The comparison includes variants of our method adopting distinct selection strategies: always select the large model, always select the small models, randomly select between the large and small models (random), and employ our advanced reinforcement learning-based model selection strategy (selection)



(a) (b)

Fig. 5 Relationship between the model selector's outcome and the relative performance. The 'Relative Score' compares the small models to the LMM; a score below 0.5 means the small model performs better. The 'Probability' reflects the chance of choosing the small model. This demonstrates our model selector's capability to discern the more effective model. (a) Vizwiz; (b) Blmm

selector and the relative scores of small models for all utilized basic visual functions, as shown in Fig. 5. This probability is written as N_s in Eq. (2). The data reveals a positive correlation between the probability N_s and the scores of small models relative to those of large models, demonstrating that our model selector has successfully learned an effective strategy for model selection.

5 Conclusion

To patch the visual ability of large multimodal models in basic visual functions, we propose a novel method that collaborates LMMs with small task-specific models. Our method introduces an automatic decision-making mechanism to determine when to invoke small models based on the input query, enabling LMMs to achieve higher accuracy and efficiency across a wide range of vision-language tasks. Extensive experiments on established benchmarks, including MM-Vet, LLaVA-Bench, VizWiz, and our custom Blmm dataset, demonstrate significant improvements in the visual capabilities of LMMs. This collaborative approach not only mitigates specific limitations of LMMs but also provides a flexible and scalable framework for task-specific optimization. Future work will focus on deeper integration of LMMs with small models and extend the application of this method to more dynamic, real-world scenarios.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 62495082 and U2336213).

Competing interests

Shiguang SHAN is an Editorial Board member of the journal and a co-author of this article. To minimize bias, he was excluded from all editorial decision-making related to the acceptance of this article for publication. The remaining authors declare no conflict of interest.

Appendixes

The Appendices provide additional details to support the main content of this paper, including (1) the specifications of the task-specific models used in our method for each visual function, (2) detailed textual instructions used in our method, (3) supplementary visual examples to illustrate key results, (4) additional experimental results including inference speed, and (5) a comprehensive description of our Blmm dataset, to further validate the efficiency and robustness of our method.

Appendix A Task-specific models employed in our method

Each visual function corresponds to a small task-specific model, all of which are detailed in Table A1. Our method allows for the arbitrary choice of small models, including those that are custom-made. In this paper, the small models we have chosen are all off-the-shelf. Next, we introduce each of the small models we used. Links to the model pages are provided at the end of each paragraph.

DETR-R50 The DETR model [44] (End-to-End Object Detection with Transformers), equipped with a ResNet-50 backbone, employs object queries to identify objects within an image. This model was trained on the COCO 2017 [56] dataset and the number of queries is configured to be 100. Model Page

Table A1 The small task-specific model used for each visual function

Visual function	Model name	#params
Object detecion	DETR-R50 [44]	41M
Semantic segmentation	Mask2Former-SwinL [45]	216M
Visual grounding	GroundingDINO-SwinT [46]	172M
Image caption	Blip-Image-Caption-Base [47]	247M
Cash recognition	API from Baidu Brain [48]	–
Optical character recognition	PaddleOCR [49]	15.8M
Face expression recognition	Vit-Face-Expression [50]	85.8M
Action recognition	Human-Action-Recognition [51]	85.8M
Dish recognition	API from Baidu Brain [52]	–
Fruit vegitable recognition	API from Baidu Brain [53]	–
Tactile paving segmentation	GRFB-UNet [54]	20.5M
Traffic light recognition	PTL-LytNet [55]	2M
Outdoors semantic segmentation	Mask2Former-SwinL [45]	216M

Mask2Former-SwinL By integrating multi-scale deformable attention and masked attention mechanisms, Mask2Former [45] effectively tackles instance, semantic, and panoptic segmentation challenges within a unified framework. In our study, we utilize two versions of the Mask2Former model for different scenes: the swin-large version trained on ADE-20K [57] for Semantic Segmentation and the Swin-Large version trained on Mapillary-Vistas [58] for outdoors semantic segmentation.
Model Page (ADE20K),
Model Page (Mapillary Vistas)

GroundingDINO-SwinT The GroundingDINO-SwinT [46] model, utilizing the `swin-T-224-1k` backbone, is capable of performing open-set object detection, enabling it to identify arbitrary objects based on human descriptions. Model Page

Blip-Image-Caption-Base BLIP [47] unifies the understanding and generation of visual language tasks, excelling across various downstream tasks. We specifically utilize its Image Captioning functionality. Although BLIP can handle a range of language-vision tasks, it has only 247 million parameters. Model Page

API from Baidu Brain In our research, we selected three APIs from Baidu, specifically for cash recognition, dish recognition, and fruit-vegetable recognition. The cash recognition API [48] is capable of identifying a wide range of commonly used modern currencies, providing details such as the currency name, face value, and additional information. The dish recognition API [52] delivers the name of the dish and the confidence level associated with the image's analysis. Meanwhile, the fruit vegetable recognition API [53] determines the number of fruits and vegetables present and identifies their respective names. API Page

PaddleOCR PaddleOCR [49] is an open-source optical character recognition (OCR) toolkit developed by PaddlePaddle. We utilize the

PP-OCRv4 model in our method. Model Page

Vit-Face-Expression The Vit-Face-Expression model has been fine-tuned from the ViT-Base model specifically for the task of recognizing facial emotions. The training dataset is FER2013 [59] dataset, which comprises approximately 30,000 images of facial expressions categorized into seven distinct emotions. Model Page

Human-Action-Recognition The Human Action Recognition model has been fine-tuned from the ViT-Base model specifically for the task of recognizing human actions. The training dataset [60] includes 12.6k images, each assigned to a single category out of 15 types of human activities. Model Page

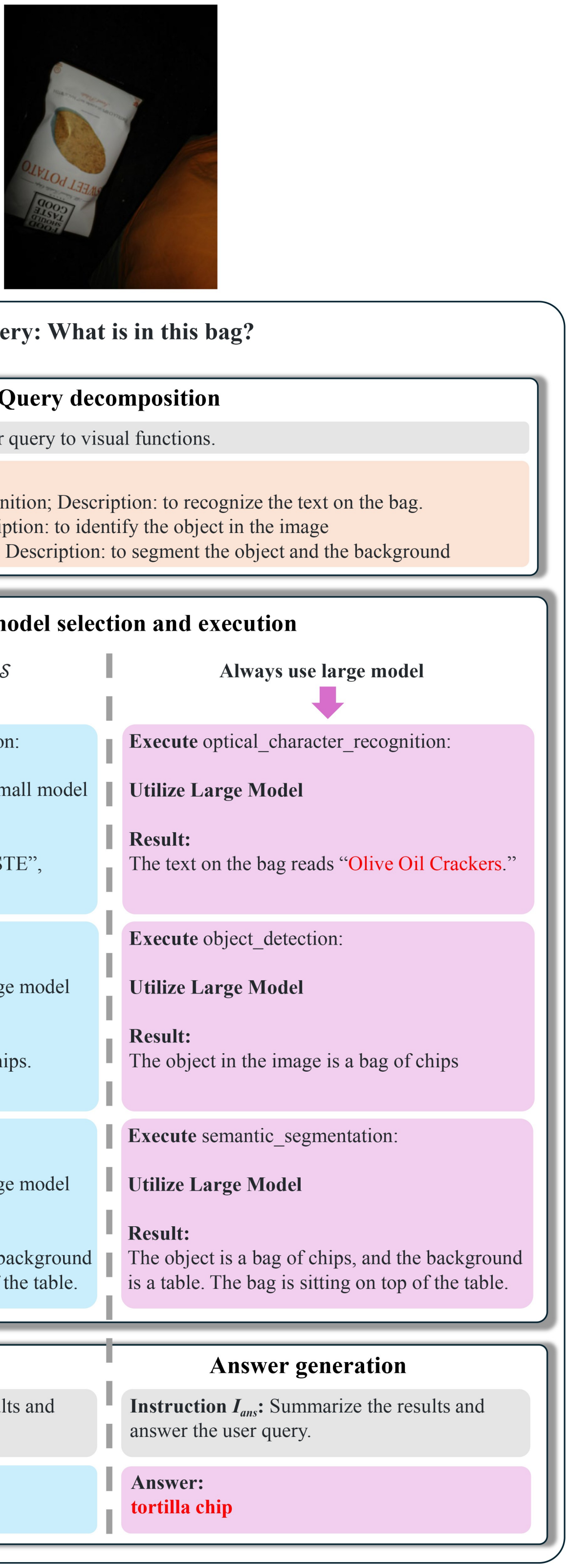
GRFB-UNet The GRFB-UNet [54] merges the UNet with a multi-

Instruction	Detailed text
I_{qry} in query decomposition	You are an AI assistant tasked with answering questions about an image. You have several capabilities at your disposal to help you answer the question. You can choose to use one or more capabilities to provide the best response. Please respond in the following JSON format: <Solution>{{"Thought": "thought", "Chain": chain}}</Solution> The "Thought" field should explain your reasoning in up to 80 words, outlining why you chose the specific capability chain. The "Chain" field should consist of a list of capabilities you plan to use, with each entry in the following format: {{"name": "capability_name", "target": "explanation of why this capability is chosen, as accurate and short as possible."}} The "target" field is important. You can fill it like "find all objects", "look for the red cat", "detect texts" ... You should select no more than 3 capabilities to use. So you should choose the most suitable capabilities to achieve the target. Below is a list of all available capabilities in a "name: description" format: {capabilities} The User Question is: {user_question} Please choose the capabilities that you want to use to answer the question. You can choose one or more capabilities. You can also choose not to use any capabilities to answer the question. Pay attention to the order of capabilities. You should always respond in the following format: <Solution> `SOLUTION` </Solution> `SOLUTION` should strictly comply with JSON format described above.
$I_{\phi i}$ in visual model execution	The target: {capability_target} You need to achieve the target and generate the response.
I_{ans} in answer generation	You are a blind visual question and answer assistance system. Your task is to answer a user question based on the input image. {tool execution results} Now you need to answer a user question `{user_question}`. You can use the original image and the additional information above to answer the question. Notice that the additional information is generated by the neural network, which may be wrong. Please respond in the following JSON format: <Solution>{{"Thought": "thought", "Answer": "answer"}}</Solution> In the "Thought" field, provide a concise explanation within 80 words, outlining the reasoning behind your answer. The "Answer" field should contain the final answer to the question. No additional information or advice is required. Just answer the question. You should always respond in the following format: <Solution> `SOLUTION` </Solution> `SOLUTION` should strictly comply with JSON format described above. Now you can start to answer the question `{user_question}`.

Fig. A1 The detailed instructions used in our method

scale feature extraction technique known as the group receptive field block to identify the shape and location of tactile paving. The model is trained on a self-constructed TP dataset and has 20.5 million parameters. Model Page

PTL-LytNet The PTL-LytNet [55], a modification of MobileNet, is designed to classify different states of pedestrian traffic lights and



Query: What is in this bag?

Query decomposition

Instruction I_{qry}: Decompose the user query to visual functions.

LLaVA output:
1. Function: optical_character_recognition; Description: to recognize the text on the bag.
2. Function: object_detection; Description: to identify the object in the image
3. Function: semantic_segmentation; Description: to segment the object and the background

Visual model selection and execution

Utilize our model selector $\mathcal{S}$	Always use large model
Execute optical_character_recognition: **Model Selector:** $\mathcal{F}_i = \mathcal{M}_i$, use the small model **Result**: “SWEETPOTATO”, “GOOD”, “TASTE”, “CINOHS”, “FOOD”, …	**Execute** optical_character_recognition: **Utilize Large Model** **Result:** The text on the bag reads “Olive Oil Crackers.”
Execute object_detection: **Model Selector:** $\mathcal{F}_i = \mathcal{L}$, use the large model **Result:** The object in the image is a bag of chips.	**Execute** object_detection: **Utilize Large Model** **Result:** The object in the image is a bag of chips
Execute semantic_segmentation: **Model Selector:** $\mathcal{F}_i = \mathcal{L}$, use the large model **Result:** The object is a bag of chips, and the background is a table. The bag is sitting on top of the table.	**Execute** semantic_segmentation: **Utilize Large Model** **Result:** The object is a bag of chips, and the background is a table. The bag is sitting on top of the table.

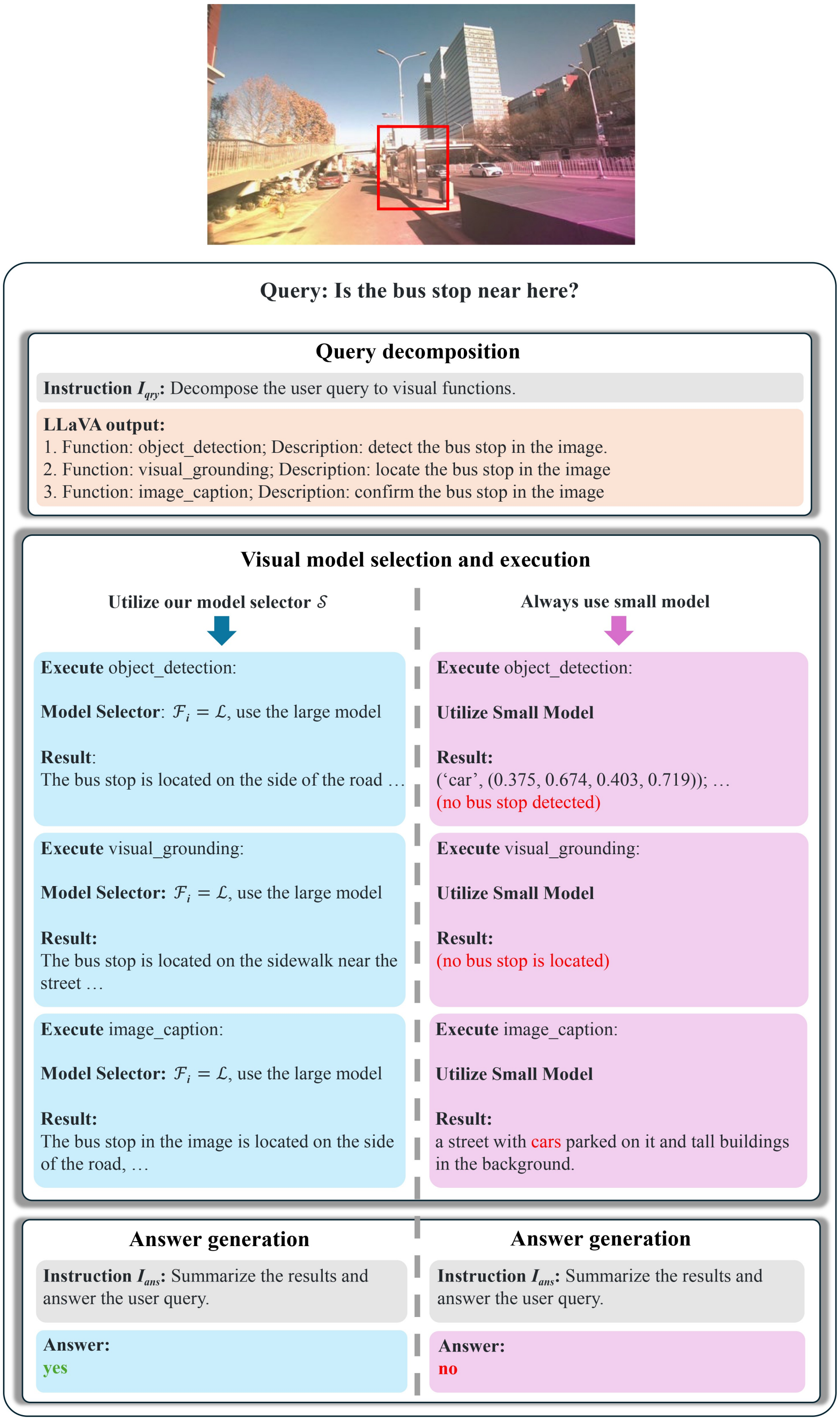
Answer generation	Answer generation
Instruction I_{ans}: Summarize the results and answer the user query.	**Instruction I_{ans}:** Summarize the results and answer the user query.
Answer: **Sweet potato chips**	**Answer:** **tortilla chip**

Fig. A2 An example of our method during testing. The left side depicts the execution process of our method, while the right side demonstrates the results that would be obtained if alternative models were chosen. The correct and incorrect results are shown in green and red respectively

the locations of crosswalks. Even with only 2 million parameters, it can accurately perform recognition under various weather conditions. Model Page

Appendix B Detailed instructions in our method

We list the detailed instructions in Fig. A1, which are utilized to direct the LMM in accomplishing different tasks. The instruction



Query: Is the bus stop near here?

Query decomposition

Instruction I_{qry}: Decompose the user query to visual functions.

LLaVA output:
1. Function: object_detection; Description: detect the bus stop in the image.
2. Function: visual_grounding; Description: locate the bus stop in the image
3. Function: image_caption; Description: confirm the bus stop in the image

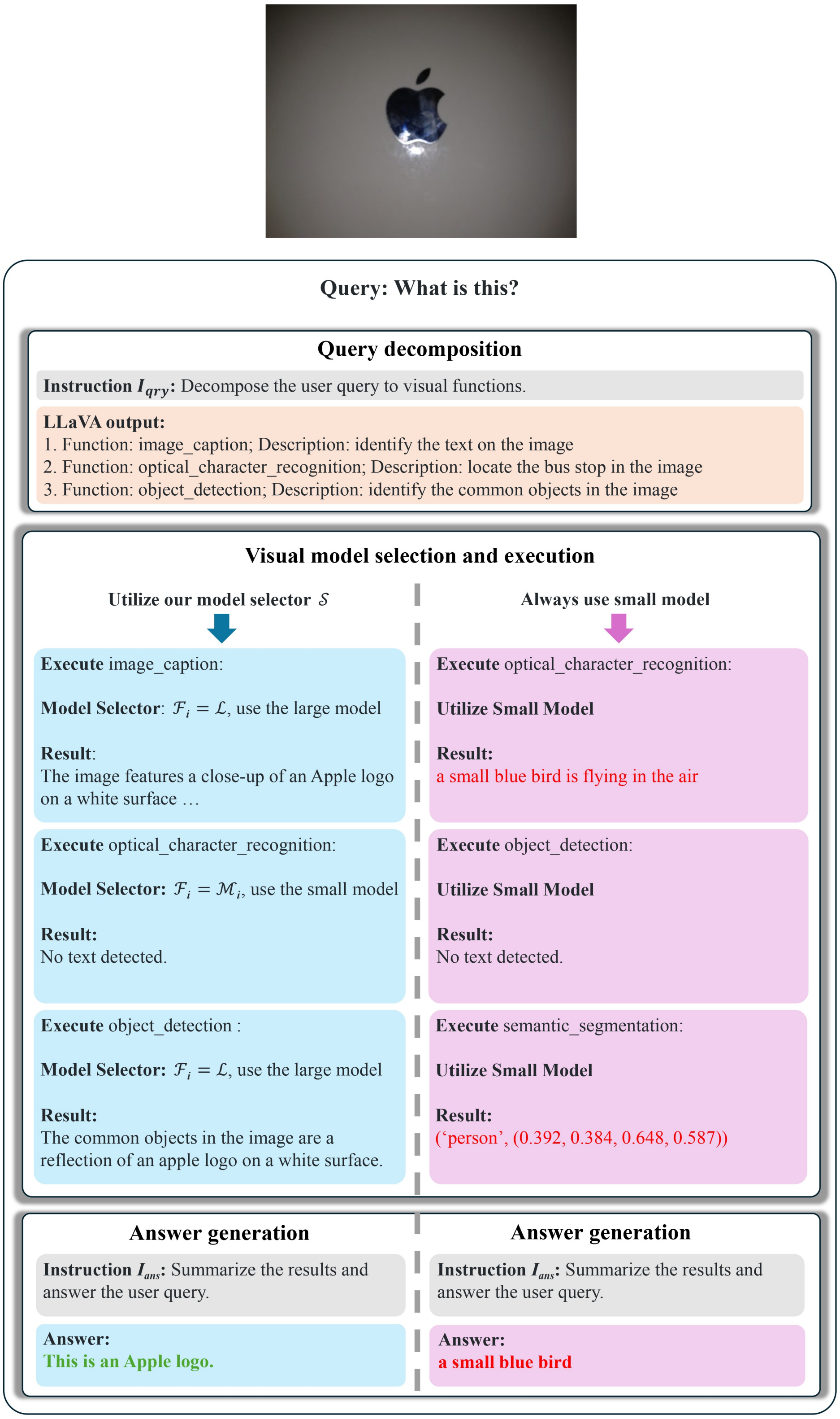
Visual model selection and execution

Utilize our model selector $\mathcal{S}$	Always use small model
Execute object_detection: **Model Selector**: $\mathcal{F}_i = \mathcal{L}$, use the large model **Result**: The bus stop is located on the side of the road …	**Execute** object_detection: **Utilize Small Model** **Result:** ('car', (0.375, 0.674, 0.403, 0.719)); … (no bus stop detected)
Execute visual_grounding: **Model Selector:** $\mathcal{F}_i = \mathcal{L}$, use the large model **Result:** The bus stop is located on the sidewalk near the street …	**Execute** visual_grounding: **Utilize Small Model** **Result:** (no bus stop is located)
Execute image_caption: **Model Selector:** $\mathcal{F}_i = \mathcal{L}$, use the large model **Result:** The bus stop in the image is located on the side of the road, …	**Execute** image_caption: **Utilize Small Model** **Result:** a street with cars parked on it and tall buildings in the background.

Answer generation	**Answer generation**
Instruction I_{ans}: Summarize the results and answer the user query.	**Instruction I_{ans}:** Summarize the results and answer the user query.
Answer: **yes**	**Answer:** **no**

Fig. A3 An example of our method during testing. The left side depicts the execution process of our method, while the right side demonstrates the results that would be obtained if alternative models were chosen. The correct and incorrect results are shown in green and red respectively

I_{qry} in the Query Decomposition phase is used to decompose the user query into basic visual functions through LMM. The instruction I_{ϕ_i} in the Visual Model Execution phase is used to guide the LMM to perform a specific visual function. The instruction I_{ans} in the Answer Generation phase is used to summarize all the perceptual results and generate an answer to the user query. The variables



Query: What is this?

Query decomposition

Instruction I_{qry}: Decompose the user query to visual functions.

LLaVA output:
1. Function: image_caption; Description: identify the text on the image
2. Function: optical_character_recognition; Description: locate the bus stop in the image
3. Function: object_detection; Description: identify the common objects in the image

Visual model selection and execution

Utilize our model selector $\mathcal{S}$

Execute image_caption:

Model Selector: $\mathcal{F}_i = \mathcal{L}$, use the large model

Result:
The image features a close-up of an Apple logo on a white surface …

Execute optical_character_recognition:

Model Selector: $\mathcal{F}_i = \mathcal{M}_i$, use the small model

Result:
No text detected.

Execute object_detection :

Model Selector: $\mathcal{F}_i = \mathcal{L}$, use the large model

Result:
The common objects in the image are a reflection of an apple logo on a white surface.

Always use small model

Execute optical_character_recognition:

Utilize Small Model

Result:
a small blue bird is flying in the air

Execute object_detection:

Utilize Small Model

Result:
No text detected.

Execute semantic_segmentation:

Utilize Small Model

Result:
('person', (0.392, 0.384, 0.648, 0.587))

Answer generation

Instruction I_{ans}: Summarize the results and answer the user query.

Answer:
This is an Apple logo.

Answer generation

Instruction I_{ans}: Summarize the results and answer the user query.

Answer:
a small blue bird

Fig. A4 An example of our method during testing. The left side depicts the execution process of our method, while the right side demonstrates the results that would be obtained if alternative models were chosen. The correct and incorrect results are shown in green and red respectively

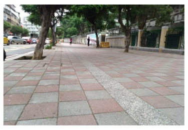
wrapped in braces will be replaced based on user input and intermediate results.

Appendix C Additional experimental results

In this section, we present additional experimental results, including visualized examples from the Blmm and VizWiz datasets, as well as the inference speed of our method.

Visualized results: Figures A2 to A4 illustrate the execution process of our method during testing. In each figure, the left side illustrates the workflow of our proposed method, while the right side showcases the results obtained using alternative model selection strategies. These examples show the effectiveness of our model selector. We also provide a video demonstration in this link that shows our method in action.

Figure A5 illustrates the responses of our method compared to other methods across multiple cases. Our method demonstrates performance that surpasses other methods. Due to the lack of tool-using capabilities, LLaVA [1] sometimes fails to provide correct answers. As GPT4Tools [24] utilizes LLaMA [18], which inherently lacks visual capabilities and the tools are limited in visual capability, it is unable to answer some questions. LLaVA-Plus [27] might have compromised the model's original language capabilities during fine-

	 Could you describe the denomination of this bill for me?	 **Is there a blind path in the picture?**	 **Am I facing the zebra crossing straight now?**
LLaVA	The bill is worth 2005 Chinese Yuan.	Yes, the path is safe for blind individuals to walk on …	Yes
GPT4Tools	The denomination of the bill in the image is 100 Yuan.	No, there is no blind path in the picture.	I'm sorry, but as an AI language model, I don't have the ability …
LLaVA-Plus	I'm sorry, but I cannot provide an answer to the question …	The image features a small, safe path that appears to be a blind path…	The image shows a person …
Ours	The denomination of the bill is 10 CNY …	No	No

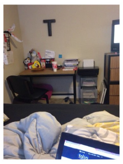
	 **What is the letter on the wall?**	 **What does the sky look like?**	 **What is this?**
LLaVA	C	The sky looks cloudy and overcast, with no blue sky or sunshine visible.	Mailbox
GPT4Tools	The letter on the wall is "t".	I'm sorry, …	The image you provided is a blue mailbox with a white sticker on it.
LLaVA-Plus	The image does not provide any information about the year 2023.	I'm sorry, but I cannot provide an answer to the question …	Sure! I use an image generation model stable-diffusion to generate image on your prompt.
Ours	The letter on the wall is 'T'.	The sky looks cloudy and overcast …	This is a United States Postal Service mailbox.

Fig. A5 Some examples of our method and the comparative methods. The correct and incorrect results are shown in green and red respectively

Table A2 Inference latency comparison on a single RTX 4090

Metric	VisGPT	LLaVA	GPT4Tools	CLLM	LLaVA-Plus	Ours
Latency/s	3.0	6.9	2.1	31.0	27.6	20.3

tuning, leading to some incorrect responses.

Inference speed: To evaluate the efficiency of our method, we conducted experiments on a single RTX 4090 GPU and summarized the results in Table A2. Our method's inference latency is approximately equivalent to three calls to LLaVA, making it competitive among methods that utilize both large models and external tools.

For comparison, VisGPT directly queries GPT APIs with minimal delay, while GPT4Tools leverages LLaMA and avoids processing image tokens, resulting in lower latency. LLaVA, as a purely vision-language model without external tools, is the second fastest due to its single inference process. Methods such as CLLM, LLaVA-Plus, and ours involve both large models and external small models, which inherently increases latency due to additional interfaces. However, the latency differences among these methods are minimal, reflecting their similar workflow complexity.

Appendix D Details of our Blmm dataset

Our Blmm dataset is specifically designed to address the needs of visually impaired individuals. While it shares some similarities with VizWiz [42], it places a stronger emphasis on evaluating a broader range of visual capabilities. The VizWiz dataset primarily focuses on four types of questions: object detection and recognition, color recognition, text recognition, and object counting. However, it does not comprehensively cover all real-world scenarios encountered by visually impaired individuals and evaluates only a limited set of visual abilities.

In contrast, Blmm is designed to include more complex questions and diverse outdoor scenes, making it more challenging and better aligned with real-world needs. Unlike VizWiz, which predominantly features simpler queries (most samples have single-word answers), Blmm emphasizes richer and more varied interactions. Furthermore, compared to general VQA benchmarks such as LLaVA-Bench and MM-Vet, Blmm is more tailored to the specific requirements of visually impaired users. LLaVA-Bench and MM-Vet involve synthetic or artificial images and focus heavily on logical reasoning and detailed description tasks, resulting in longer response lengths. However, these datasets are less grounded in real-world user needs compared to Blmm and VizWiz. For a comprehensive comparison of these datasets, including their key characteristics and differences, please refer to Table A3.

Table A3 Information of datasets. Ind., Outd., and Art. stand for indoor, outdoor, and artificial images respectively

	LLaVA-Bench	MM-Vet	VizWiz	Blmm
Resolution	438	779	1032	660
#Sample	90	218	537	195
Scenes	Ind., Outd.	Ind., Outd., Art.	Ind.	Ind., Outd.
Answer length	405.0	57.2	2	11.6

To accurately identify the needs of visually impaired individuals, we conducted a review of related work and engaged in direct conversations with 17 visually impaired individuals. As a result, we compiled a list of six scenarios: work, reading, socializing, entertainment, dining, and travel. The evaluated visual abilities increased to 13 types, such as food recognition, blind path recognition, traffic light recognition, and more.

The visual questions and answers in our dataset are derived from scenarios commonly faced by visually impaired individuals in their daily lives. For the collection of images, we employed a dual approach, combining both photographs we took ourselves and images downloaded from the Internet, to ensure diversity and breadth. For instance, for outdoor activities, we captured numerous photos using a D455 camera and also sourced some from the Internet. Every image in our dataset has been carefully examined to ensure it does not contain any personal privacy information. We aimed for a balanced representation across different categories. Following the image collection, we meticulously crafted corresponding questions and answers for each image, limiting to no more than two questions per image. The design of these questions and answers was carefully tailored to align with the communication patterns of visually impaired individuals, prioritizing clarity and brevity.

The dataset comprises 647 question-answer pairs, divided into training and validation splits. The training split contains 452 question-answer pairs, while the validation split includes 195 pairs. The Blmm dataset and evaluation code are released in this link.

References

[1] Liu H, Li C, Li Y, Lee Y J. Improved baselines with visual instruction tuning. In: Proceedings of 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024, 26296−26306

[2] OpenAI, Achiam J, Adler S, Agarwal S, Ahmad L, et al. GPT-4 technical report. 2023, arXiv preprint arXiv: 2303.08774

[3] Yang Z, Li L, Lin K, Wang J, Lin C C, Liu Z, Wang L. The dawn of LMMs: preliminary explorations with GPT-4V (ision). 2023, arXiv preprint arXiv: 2309.17421

[4] Liu Y, Li Z, Huang M, Yang B, Yu W, Li C, Yin X C, Liu C L, Jin L, Bai X. OCRBench: on the hidden mystery of OCR in large multimodal models. Science China Information Sciences, 2024, 67(12): 220102

[5] Wu C, Yin S, Qi W, Wang X, Tang Z, Duan N. Visual ChatGPT: talking, drawing and editing with visual foundation models. 2023, arXiv preprint arXiv: 2303.04671

[6] Schick T, Dwivedi-Yu J, Dessí R, Raileanu R, Lomeli M, Hambro E, Zettlemoyer L, Cancedda N, Scialom T. Toolformer: language models can teach themselves to use tools. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 2997

[7] OpenAI. Introducing ChatGPT. See openai.com/blog/chatgpt/ website, 2022

[8] Touvron H, Lavril T, Izacard G, Martinet X, Lachaux M A, Lacroix T, Rozière B, Goyal N, Hambro E, Azhar F, Rodriguez A, Joulin A, Grave E, Lample G. LLaMA: open and efficient foundation language models. 2023, arXiv preprint arXiv: 2302.13971

[9] Touvron H, Martin L, Stone K, Albert P, Almahairi A, et al. Llama 2: open foundation and fine-tuned chat models. 2023, arXiv preprint arXiv: 2307.09288

[10] Zeng A, Liu X, Du Z, Wang Z, Lai H, Ding M, Yang Z, Xu Y, Zheng W, Xia X, Tam W L, Ma Z, Xue Y, Zhai J, Chen W, Liu Z, Zhang P, Dong Y, Tang J. GLM-130B: an open bilingual pre-trained model. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[11] The Vicuna Team. Vicuna: an open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. See lmsys.org/blog/2023-03-30-vicuna/ website, 2023

[12] Alayrac J B, Donahue J, Luc P, Miech A, Barr I, et al. Flamingo: a visual language model for few-shot learning. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022, 23716−23736

[13] Zhu D, Chen J, Shen X, Li X, Elhoseiny M. MiniGPT-4: enhancing vision-language understanding with advanced large language models. In: Proceedings of the 12th International Conference on Learning Representations. 2024

[14] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N. An image is worth 16x16 words: transformers for image recognition at scale. In: Proceedings of the 9th International Conference on Learning Representations. 2021

[15] Li J, Li D, Savarese S, Hoi S. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In: Proceedings of the 40th International Conference on Machine Learning. 2023, 814

[16] Gemini Team Google, Anil R, Borgeaud S, Alayrac J B, Yu J, et al. Gemini: a family of highly capable multimodal models. 2023, arXiv preprint arXiv: 2312.11805

[17] Li K, He Y, Wang Y, Li Y, Wang W, Luo P, Wang Y, Wang L, Qiao Y. VideoChat: chat-centric video understanding. 2023, arXiv preprint arXiv: 2305.06355

[18] Li Y, Wang C, Jia J. LLaMA-VID: an image is worth 2 tokens in large language models. In: Proceedings of the 18th European Conference on Computer Vision. 2025, 323−340

[19] Rubenstein P K, Asawaroengchai C, Nguyen D D, Bapna A, Borsos Z, et al. AudioPaLM: a large language model that can speak and listen. 2023, arXiv preprint arXiv: 2306.12925

[20] Girdhar R, El-Nouby A, Liu Z, Singh M, Alwala K V, Joulin A, Misra I. ImageBind one embedding space to bind them all. In: Proceedings of 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, 15180−15190

[21] Shen Y, Song K, Tan X, Li D, Lu W, Zhuang Y. HuggingGPT: solving AI tasks with ChatGPT and its friends in hugging face. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 38154−38180

[22] Thoppilan R, De Freitas D, Hall J, Shazeer N, Kulshreshtha A, et al. LaMDA: language models for dialog applications. 2022, arXiv preprint arXiv: 2201.08239

[23] Gao L, Madaan A, Zhou S, Alon U, Liu P, Yang Y, Callan J, Neubig G. PAL: program-aided language models. In: Proceedings of the 40th International Conference on Machine Learning. 2023, 10764−10799

[24] Yang R, Song L, Li Y, Zhao S, Ge Y, Li X, Shan Y. GPT4Tools: teaching large language model to use tools via self-instruction. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 3149

[25] Yang Z, Li L, Wang J, Lin K, Azarnasab E, Ahmed F, Liu Z, Liu C, Zeng M, Wang L. MM-REACT: prompting ChatGPT for multimodal reasoning and action. 2023, arXiv preprint arXiv: 2303.11381

[26] Weng Y, He S, Liu K, Liu S, Zhao J. ControlLM: crafting diverse personalities for language models. 2024, arXiv preprint arXiv: 2402.10151

[27] Liu S, Cheng H, Liu H, Zhang H, Li F, Ren T, Zou X, Yang J, Su H, Zhu J, Zhang L, Gao J, Li C. LLaVA-Plus: learning to use tools for creating multimodal agents. In: Proceedings of the 18th European Conference on Computer Vision. 2025, 126−142

[28] Yang Z, Chen G, Li X, Wang W, Yang Y. DoraemonGPT: toward understanding dynamic scenes with large language models (Exemplified as a video agent). In: Proceedings of the 41st International Conference on Machine Learning. 2024, 55976−55997

[29] Yang Y, Zhuang Y, Pan Y. Multiple knowledge representation for big data artificial intelligence: framework, applications, and case studies. Frontiers of Information Technology & Electronic Engineering, 2021, 22(12): 1551−1558

[30] Quan R, Dong X, Wu Y, Zhu L, Yang Y. Auto-ReID: searching for a part-aware ConvNet for person re-identification. In: Proceedings of 2019 IEEE/CVF International Conference on Computer Vision. 2019, 3750−3759

[31] Liu H, Simonyan K, Yang Y. DARTS: differentiable architecture search. In: Proceedings of the 7th International Conference on Learning Representations. 2019

[32] Chung H W, Hou L, Longpre S, Zoph B, Tai Y, et al. Scaling instruction-finetuned language models. Journal of Machine Learning Research, 2024, 25(70): 1−53

[33] Iyer S, Lin X V, Pasunuru R, Mihaylov T, Simig D, Yu P, Shuster K, Wang T, Liu Q, Koura P S, Li X, O’Horo B, Pereyra G, Wang J, Dewan C, Celikyilmaz A, Zettlemoyer L, Stoyanov V. OPT-IML: scaling language model instruction meta learning through the lens of generalization. 2022, arXiv preprint arXiv: 2212.12017

[34] Wang Y, Kordi Y, Mishra S, Liu A, Smith N A, Khashabi D, Hajishirzi H. Self-instruct: aligning language models with self-generated instructions. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023, 13484−13508

[35] Singhal K, Azizi S, Tu T, Mahdavi S S, Wei J, Chung H W, Scales N, Tanwani A, Cole-Lewis H, Pfohl S, Payne P, Seneviratne M, Gamble P, Kelly C, Babiker A, Schärli N, Chowdhery A, Mansfield P, Demner-Fushman D, Agüera Y arcas B, Webster D, Corrado G S, Matias Y, Chou K, Gottweis J, Tomasev N, Liu Y, Rajkomar A, Barral J, Semturs C,

Karthikesalingam A, Natarajan V. Large language models encode clinical knowledge. Nature, 2023, 620(7972): 172−180

[36] Taylor R, Kardas M, Cucurull G, Scialom T, Hartshorn A, Saravia E, Poulton A, Kerkez V, Stojnic R. Galactica: a large language model for science. 2022, arXiv preprint arXiv: 2211.09085

[37] Wu S, Irsoy O, Lu S, Dabravolski V, Dredze M, Gehrmann S, Kambadur P, Rosenberg D, Mann G. BloombergGPT: a large language model for finance. 2023, arXiv preprint arXiv: 2303.17564

[38] Wang W, Wei F, Dong L, Bao H, Yang N, Zhou M. MINILM: deep self-attention distillation for task-agnostic compression of pre-trained transformers. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 485

[39] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal policy optimization algorithms. 2017, arXiv preprint arXiv: 1707.06347

[40] Liu H, Li C, Wu Q, Lee Y J. Visual instruction tuning. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 1516

[41] Yu W, Yang Z, Li L, Wang J, Lin K, Liu Z, Wang X, Wang L. MM-Vet: evaluating large multimodal models for integrated capabilities. In: Proceedings of the 41st International Conference on Machine Learning. 2024, 57730−57754

[42] Gurari D, Li Q, Stangl A J, Guo A, Lin C, Grauman K, Luo J, Bigham J P. VizWiz grand challenge: answering visual questions from blind people. In: Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2018, 3608−3617

[43] Goyal Y, Khot T, Summers-Stay D, Batra D, Parikh D. Making the V in VQA matter: elevating the role of image understanding in visual question answering. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017, 6904−6913

[44] Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S. End-to-end object detection with transformers. In: Proceedings of the 16th European Conference on Computer Vision. 2020, 213−229

[45] Cheng B, Misra I, Schwing A G, Kirillov A, Girdhar R. Masked-attention mask transformer for universal image segmentation. In: Proceedings of 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, 1290−1299

[46] Liu S, Zeng Z, Ren T, Li F, Zhang H, Yang J, Jiang Q, Li C, Yang J, Su H, Zhu J, Zhang L. Grounding DINO: marrying DINO with grounded pre-training for open-set object detection. In: Proceedings of the 18th European Conference on Computer Vision. 2025, 38−55

[47] Li J, Li D, Xiong C, Hoi S C H. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 12888−12900

[48] Baidu. Cash recognition API document. See ai.baidu.com/ai-doc/IMAGERECOGNITION/pk3bcxavy website, 2024

[49] PaddleOCR. PaddleOCR project. See github.com/PaddlePaddle/PaddleOCR website, 2024

[50] Trpakov. Vision transformer (ViT) for facial expression recognition model card. See huggingface.co/trpakov/vit-face-expression website, 2024

[51] Rvv-karma. Human action recognition ViT model card. See huggingface.co/rvv-karma/Human-Action-Recognition-VIT-Base-patch16-224 website, 2024

[52] Baidu. Dish recognition API document. See ai.baidu.com/ai-doc/IMAGERECOGNITION/tk3bcxbb0 website, 2024

[53] Baidu. Fruit and vegetable recognition API document. See ai.baidu.com/ai-doc/IMAGERECOGNITION/wk3bcxevq website, 2024

[54] Zhang X, Liang L, Zhao S, Wang Z. GRFB-UNet: a new multi-scale attention network with group receptive field block for tactile paving segmentation. Expert Systems with Applications, 2024, 238: 122109

[55] Yu S, Lee H, Kim J. LYTNet: a convolutional neural network for real-time pedestrian traffic lights and zebra crossing recognition for the visually impaired. In: Proceedings of the 18th International Conference on Computer Analysis of Images and Patterns. 2019, 259−270

[56] Lin T Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick C L. Microsoft COCO: common objects in context. In: Proceedings of the 13th European Conference on Computer Vision. 2014, 740−755

[57] Zhou B, Zhao H, Puig X, Fidler S, Barriuso A, Torralba A. Scene parsing through ADE20K dataset. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017, 633−641

[58] Neuhold G, Ollmann T, Rota Bulò S, Kontschieder P. The mapillary vistas dataset for semantic understanding of street scenes. In: Proceedings of 2017 IEEE International Conference on Computer Vision. 2017, 4990−4999

[59] Goodfellow I J, Erhan D, Luc Carrier P, Courville A, Mirza M, Hamner B, Cukierski W, Tang Y, Thaler D, Lee D H, Zhou Y, Ramaiah C, Feng F, Li R, Wang X, Athanasakis D, Shawe-Taylor J, Milakov M, Park J, Ionescu R, Popescu M, Grozea C, Bergstra J, Xie J, Romaszko L, Xu B, Chuang Z, Bengio Y. Challenges in representation learning: a report on three machine learning contests. Neural Networks, 2015, 64: 59−63

[60] Kaggle. Human action recognition (HAR) dataset. See www.kaggle.com/datasets/meetnagadia/human-action-recognition-har-dataset website, 2022

Hao LIANG received the BS degree in computer science and technology from Peking University, China in 2021. He is currently pursuing the PhD degree with the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), China. His research interests include object detection and active vision.

Xiaolong ZHANG received the BS degree in computer science and technology from Dalian University of Technology, China in 2022. He is currently pursuing his master's degree with the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), China. His research mainly focuses on general visual perception.

Meina KAN received the PhD degree from the University of Chinese Academy of Sciences, China. She is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences (CAS), China. Her research interests include machine vision especially embodied AI, transfer learning, and deep learning.

Shiguang SHAN is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences (CAS), China and the Director of the Key Laboratory of Intelligent Information Processing, CAS, China. He has published more than 350 articles in related areas. He served as the General Co-Chair for the IEEE FG 2023 and ACCV 2022 and the Area Chair for tens of international conferences, including CVPR, ICCV, ECCV, NeurIPS, ICML, AAAI, IJCAI, ACCV, ICPR, FG, and WACV. He was/is an Associate Editor of several journals, including IEEE Transactions on Image Processing, Neurocomputing, Computer Vision and Image Understanding, Transactions on Machine Learning Research, and Pattern Recognition Letters.

Xilin CHEN is currently a professor with the Institute of Computing Technology, Chinese Academy of Sciences, China. His research interests include image processing, computer vision, pattern recognition, and machine learning. He has authored one book and more than 400 articles in refereed journals and proceedings in related areas. He is a fellow of ACM, IAPR, and CCF. He is also an Information Sciences Editorial Board Member of Fundamental Research, an Editorial Board Member of Research, a Senior Editor of the Journal of Visual Communication and Image Representation, and the Associate Editor-in-Chief of Chinese Journal of Computers and Pattern Recognition and Artificial Intelligence. He served as the General Co-Chair for FG 2013/FG 2018 and VCIP 2022, the Program Co-Chair for ICMI 2010/FG 2024, and the Area Chair for ICCV/CVPR/ECCV/NeurIPS for more than ten times.